Lessons from My Time @Google

Ryan Scranton

Personal reflections on transitioning from astronomy to industry

Astronomy Has a Huge Bus Problem

- Code written by small numbers of developers
- Poor/non-existent coding standards
- Too much reliance on institutional memory
- Lack of testing/validation

Most astronomers write bad code because they've never been exposed to good code or coding practices.

Lesson 1: Code Reviews

Every line of code at Google is reviewed prior to check-in

This ensures that

- More than one person knows what the code does
- Coding standards are maintained

This encourages

- Coders to refrain from checking in code that is poorly documented and/or a complete hack
- Coders to check in smaller chunks of code more often

This does not

- Slow down development
- Ensure that the code is actually doing what is intended

Mark Chu-Carroll on Code Reviews

Lesson 2: Coding Standards

Every language has a style guide that is universal

- All code that is checked in has to be approved by someone with "readability" in that language & be lint-free
- Any engineer can look at any piece of code and figure out what it does.
- Standardized code makes you less susceptible to group members leaving.
- Your code is not a unique snowflake, particularly if no one knows how it works.

Google Code Style Guides

Other Lessons

- **Unit Tests**
  - Write your unit tests after your .h file, not your .cc file.
  - Unit tests should cover small, medium and large cases.
  - Code isn't checked in without unit tests.
- **Compilers** are smarter than you.
- **IDE**s are actually useful.
- Do your students a favor and force them to write good code that you look at. Also, make sure that they know how to properly code in **OO languages** (C++, python, Java). IDL will not get them a job outside of astronomy.

Open Sourced Google Tech

Google Coding Style Guides (C++, python, *cpplint*)
http://code.google.com/p/google-styleguide/

Google Unit Testing (C++)
http://code.google.com/p/googletest/

Protocol Buffers (C++, python, Java)
http://code.google.com/apis/protocolbuffers/

GFlags (C++, python)
http://code.google.com/p/google-gflags/

S2 -- Spherical geometry package (C++)
http://code.google.com/p/s2-geometry-library/